JOURNAL OF COMPUTER AND SYSTEM SCIENCES 37, 349–366 (1988)

Completeness Results for Conflict-Free Vector Replacement Systems*

RODNEY R. HOWELL AND LOUIS E. ROSIER

Department of Computer Sciences,
University of Texas at Austin, Austin, Texas 78712-1188

Received September 22, 1986; revised November 7, 1987

In this paper, we give completeness results for the reachability, containment, and equivalence problems for conflict-free vector replacement systems (VRSs). We first give an NP algorithm for deciding reachability, thus giving the first primitive recursive algorithm for this problem. Since Jones, Landweber, and Lien have shown this problem to be NP-hard, it follows that the problem is NP-complete. Next, we show as our main result that the containment and equivalence problems are $\prod_2^p$-complete, where $\prod_2^p$ is the set of all languages whose complements are in the second level of the polynomial-time hierarchy. In showing the upper bound, we first show that the reachability set has a semilinear set (SLS) representation that is exponential in the size of the problem description, but which has a high degree of symmetry. We are then able to utilize in part a strategy introduced by Huynh (concerning SLSs) to complete our upper bound proof. © 1988 Academic Press, Inc.

1. INTRODUCTION

The reachability, containment, and equivalence problems for vector replacement systems (VRSs) (or equivalently vector addition systems (VASs), vector addition systems with states (VASSs), or Petri nets) are the subject of many unanswered questions concerning computational complexity. The containment and equivalence problems are, in general, undecidable [1, 8]. However, the reachability problem is decidable [22] (see also [17]), and, for classes of VRSs (VASs, VASSs, Petri nets) whose reachability sets are effectively computable semilinear sets (SLSs), so are the containment and equivalence problems. Classes whose reachability sets are effectively computable SLSs include bounded VRSs [16], 5-dimensional VRSs (or, equivalently, 2-dimensional VASSs) [9], conflict-free VRSs [5], persistent VRSs [7, 18, 23, 25], and regular VRSs [6, 30]. The best known lower bound for the general reachability problem is exponential space [19]. For bounded VRSs, tight nonprimitive recursive upper and lower bounds have been shown for the containment and equivalence problems [4, 10, 20, 24, 26]. For 2-dimensional VASSs, the

* This work was supported in part by U.S. Office of Naval Research Grant N00014-86-K-0763 and National Science Foundation Grant CCR-8711579. A preliminary version was presented at the "14th International Colloquium on Automata, Languages, and Programming, July 1987, Karlsruhe, F.R.G."

0022-0000/88 $3.00

reachability, containment, and equivalence problems can be solved in DTIME($2^{2^{c^* n}}$) [10]. The reachability problem for conflict-free VRSs has been shown to be NP-hard [15]. The perhaps best studied class is that of symmetric VRSs. For this class, the reachability and equivalence problems have been shown to be exponential space complete [3, 13, 21]. Few other complexity results appear to be known concerning these problems.

In this paper, we show completeness results concerning conflict-free VRSs for these three problems. Conflict-free VASs were introduced by Crespi-Reghizzi and Mandrioli [5], who showed the reachability problem to be decidable for this class. Conflict-free Petri nets were later introduced by Landweber and Robertson [18], who showed that the reachability set of a conflict-free Petri net was semilinear and that the boundedness problem for this class could be solved in exponential time. Howell, Rosier, and Yen [11] then introduced conflict-free VRSs as a formalism that contains both conflict-free VASs and conflict-free Petri nets as special cases, but for which the boundedness problem retains the same computational complexity; i.e., the boundedness problem was shown to be PTIME-complete for all three classes. (As was pointed out in [11], even though there are translations between the three classes, these translations do not always preserve sharp complexity bounds.) In this paper, we follow the precedent established in [11] of showing upper bounds for VRSs, the most general of the three models, and showing lower bounds for systems which satisfy the definitions of all three models.

The main result of this paper is to show the equivalence and containment problems for conflict-free VRSs to be $\prod_2^p$-complete, where $\prod_2^p$ is the second level of the polynomial-time hierarchy (see Stockmeyer [28]). Our proof depends in part upon a strategy developed by Huynh [14] in showing the equivalence problem for semilinear sets to be in $\prod_2^p$. Given the fact that conflict-free VRSs have semilinear reachability sets [18], one might attempt to solve the problem by first providing a polynomial time translation from conflict-free VRSs to SLSs. A direct application of Huynh's results would then yield the desired upper bound. Unfortunately, we can show that such a translation must be exponential. Our strategy, instead, is to show that the reachability sets can be represented as SLSs with certain special properties. These properties then enable us to obtain the desired upper bound, in spite of the fact that the translation remains exponential. Now in his proof, Huynh used the fact that the membership problem for semilinear sets is NP-complete [12]. We therefore first show that the reachability problem for conflict-free VRSs is in NP. (Since Jones, Landweber, and Lien [15] have shown the problem to be NP-hard, it follows that the problem is NP-complete.) In order to show this, we give some properties of conflict-free VRSs that allow us to show that there is an instance of integer linear programming that has a solution iff a given reachability problem has a solution; furthermore, this instance of integer linear programming can be "guessed" in polynomial time. The resulting algorithm represents the first primitive recursive algorithm to solve this problem. Our next step is to give a SLS representation of the reachability set. We have already mentioned that this representation is exponential in the size of the problem description. On the other hand, we are able to provide a SLS representation which has a high degree of symmetry. It is this symmetry that allows us to utilize Huynh's technique in a manner that yields our upper bound. Finally, we show a matching lower bound to complete the proof of our main result.

The remainder of the paper is organized as follows. In Section 2, definitions of the terminology used in this paper are given. In Section 3, we give our results concerning the reachability problem. In Section 4, we give our main result, that the containment and equivalence problems for conflict-free VRSs are $\prod_2^P$-complete. Finally, we make some concluding remarks in Section 5.

2. Definitions

Let Z (N, R) denote the set of integers (nonnegative integers, rational numbers, respectively), and let Z^k (N^k, R^k) be the set of vectors of k integers (nonnegative integers, rational numbers), $Z^{k \times m}$ ($N^{k \times m}$, $R^{k \times m}$) be the set of $k \times m$ matrices of integers (nonnegative integers, rational numbers). For a vector $v \in Z^k$, let $v(i)$, $1 \leqslant i \leqslant k$, denote the ith component of v. For a matrix $V \in Z^{k \times m}$, let $V(i, j)$, $1 \leqslant i \leqslant k$, $1 \leqslant j \leqslant m$, denote the element in the ith row and jth column of V, and let v_j denote the jth column of V. For a given value of k, let $\mathbf{0}$ in Z^k denote the vector of k zeros (i.e., $\mathbf{0}(i) = 0$ for $i = 1, ..., k$). Now given vectors u, v, and w in Z^k we say:

- $v = w$ iff $v(i) = w(i)$ for $i = 1, ..., k$;
- $v \geqslant w$ iff $v(i) \geqslant w(i)$ for $i = 1, ..., k$;
- $v > w$ iff $v \geqslant w$ and $v \neq w$;
- $u = v + w$ iff $u(i) = v(i) + w(i)$ for $i = 1, ..., k$.

A $k \times m$ *vector replacement system* (VRS), is a triple (v_0, U, V), where $v_0 \in N^k$, $U \in N^{k \times m}$, and $V \in Z^{k \times m}$, such that for any i, j, $1 \leqslant i \leqslant k$, $1 \leqslant j \leqslant m$, $U(i, j) + V(i, j) \geqslant 0$. v_0 is known as the *start vector*, U is known as the *check matrix*, and V is known as the *addition matrix*. A column u_j of U is called a *check vector*, and a column v_j of V is called an *additional rule*. For any $x \in N^k$, we say addition rule v_j is *enabled* at x iff $x \geqslant u_j$. A sequence $\theta = \langle y_1, ..., y_n \rangle$ of rules in V is *enabled* at a vector x iff for each j, $1 \leqslant j \leqslant n$, y_j is enabled at $x + y_1 + \cdots + y_{j-1}$. If θ is enabled at v_0, we say θ is *valid* in (v_0, U, V). The *reachability set* of the VRS $\mathscr{V} = (v_0, U, V)$, denoted by $R(v_0, U, V)$ (or $R(\mathscr{V})$), is the set of all vectors z, such that $z = v_0 + y_1 + \cdots + y_n$ for some $n \geqslant 0$, where each y_j $(1 \leqslant j \leqslant n)$ is a column of V, and $\langle y_1, ..., y_n \rangle$ is valid. Let $\sigma = \langle w_0, ..., w_t \rangle$ be a sequence of vectors in N^k. If $w_0 = v_0$, and for every r, $1 \leqslant r \leqslant t$, there is a j such that $w_r = w_{r-1} + v_j$ and $w_{r-1} \geqslant u_j$, then we say $\langle w_0, ..., w_t \rangle$ is a *path* in (v_0, U, V). Let Ψ denote the *Parikh mapping*, such that if θ is a sequence of rules in V, then $\Psi(\theta) \in N^m$, and $\Psi(\theta)(j)$ is the number of occurrences of v_j in θ. Let $\delta(\theta)$ denote the displacement of θ, i.e., the sum of all the vectors in θ. We also define an extended Parikh mapping (see also [18]) Ψ^+ such that $\Psi^+(\theta) = \langle \Psi(\theta), \delta(\theta) \rangle$. If $\delta(\theta) \geqslant \mathbf{0}$ ($> \mathbf{0}$), we call θ a *loop* (*positive loop*).

A VRS (v_0, U, V) is said to be *conflict-free* iff

1. no number in U is greater than 1;
2. no row in V has more than one negative number; and
3. if $V(i, j) = -1$, then the only nonzero element in row i of U is $U(i, j)$.

Note that according to this definition, all elements of U are either 0 or 1, and V contains no number less than -1. Conflict-freedom guarantees that whenever any two rules v_j and $v_{j'}$ are enabled at a vector v, v_j is also enabled at $v + v_{j'}$. (Note that this must hold even when v is not in $R(v_0, U, V)$.) For a given $k \times m$ addition matrix V, the *minimal check matrix* is a $k \times m$ matrix U in which $U(i, j) = 1$ if $V(i, j) = -1$, and $U(i, j) = 0$ otherwise. It is easy to see that the set of $k \times m$ conflict-free VRSs with minimal check matrices is equivalent to the set of $k \times m$ conflict-free VASs (see [5]). Furthermore, there is an obvious translation from a conflict-free Petri net (see [18]) with k places and m transitions to a $k \times m$ conflict-free VRS whose addition rules have no elements larger than 1. Thus, our definition is general enough to include both previous definitions. In addition, all lower bounds shown in this paper are shown using VRSs having minimal check matrices and no elements larger than 1. Thus, all of our completeness results hold for conflict-free VRSs, conflict-free VASs, and conflict-free Petri nets.

The *reachability problem* for VRSs is to determine, for a given VRS $\mathscr{V}$ and a vector v, whether $v \in R(\mathscr{V})$. The *containment* and *equivalence problems* are to determine, for two given VRSs $\mathscr{V}$ and $\mathscr{V}'$, whether $R(\mathscr{V}) \subseteq R(\mathscr{V}')$ and whether $R(\mathscr{V}) = R(\mathscr{V}')$, respectively.

Part of our analysis involves notions from linear algebra and the theory of semilinear sets. For any vector $v_0 \in N^k$ and any finite set $P(=\{v_1, ..., v_m\}) \subseteq N^k$, the set $\mathscr{L}(v_0, P) = \{x \colon \exists k_1, ..., k_m \in N \text{ and } x = v_0 + \sum_{i=1}^{m} k_i v_i\}$ is called the *linear set* with *base* v_0 over the set of *periods* P. A finite union of linear sets is called a *semilinear set* (SLS for short). If $x = \sum_{i=1}^{m} a_i v_i$ for some $a_i, ..., a_m \in R^k$, then x is a *linear combination* of the vectors in P. If $a_i \geqslant 0$ for all i, then x is *nonnegative linear combination* of the vectors in P. If in addition for some i, $a_i > 0$, then x is a *positive linear combination* of the vectors in P.

3. The Reachability Problem

The first problem we would like to examine is the reachability problem for conflict-free VRSs. Jones, Landweber, and Lien [15] have shown this problem to be NP-hard. Although the problem is known to be decidable [5], no upper bound on its complexity has yet been shown. In order to tighten this gap, we will show the problem to be NP-complete. Our strategy is to guess an instance of integer linear programming whose solutions give Parikh maps of sequences of addition rules that lead to the desired vector. The following two lemmas will give sufficient conditions to guarantee that for every solution $\bar{x}$, there is a *valid* sequence θ such that $\Psi(\theta) = \bar{x}$.

LEMMA 3.1 (from [11]). *For any $k \times m$ conflict-free VRS $\mathscr{V} = (v_0, U, V)$ that is described by n bits, we can construct in time $O(n^{1.5})$ a path σ in which no rule in V is used more than once, such that if some rule v_r is not used in σ, then there is no path in which v_r is used.*

Proof. We construct σ as follows. First, we execute all rules enabled at v_0. Then we repeatedly cycle through U, executing all those rules which are enabled but have not yet been executed. We continue until a complete pass is made through U, during which no position increases in value. (Note that this is a sufficient condition to conclude that no new rules are enabled.) Clearly, no more than $m+1$ passes are made through U. On each pass except the last, there is at least one rule (say v_j) enabled that was not enabled the previous pass; i.e., some position (say p) which was zero in the previous pass is now positive. Furthermore, since $\mathscr{V}$ is conflict-free, if some rule subtracts from position p, that rule must be v_j. Therefore, position p must have never previously been positive. Thus, on each pass except the last some position becomes positive for the first time, so the number of passes is no more than $\min(k, m) + 1 = O(n^{0.5})$. Therefore, the entire procedure operates in time $O(n^{1.5})$.

Now suppose there is a path σ' using rules not in σ. Let v_r be the first such rule executed in σ'. Then all rules used before v_r in σ' are used in σ. Since v_r is not executed in σ, no position i such that $U(i, r) = 1$ ever decreases in value in σ; hence, if these positions ever become positive in σ, they must remain positive. Since all rules executed prior to v_r in σ' are also executed in σ, these positions must clearly become positive in σ. Then v_r is enabled by σ, a contradiction. Therefore, if v_r is not used in σ, then there is no path in which v_r is used. ∎

LEMMA 3.2. *Let (v_0, U, V) be a $k \times m$ conflict-free VRS, and let θ be an arbitrary sequence of rules from V. If every rule in θ appears in some path that uses only rules from θ, and if $\delta(\theta) + v_0 \geqslant \mathbf{0}$, then there is some valid sequence θ' such that $\Psi(\theta') = \Psi(\theta)$.*

Proof. We will construct a path σ consisting of a sequence of n *segments*, $\sigma_1, \ldots, \sigma_n$, where n is the maximum number of times any rule appears in θ. Each segment will execute a sequence containing at most one occurrence of each rule in θ. Furthermore, σ will be such that if some segment does not use some rule, then no succeeding segment will use that rule. Now, if we restrict our VRS to contain only the rules used in θ, then from Lemma 3.1, some sequence containing every rule in θ exactly once is valid at v_0. The execution of this sequence yields σ_1. We construct segment σ_r, $2 \leqslant r \leqslant n$, as follows: while there is an enabled rule v_j which occurs at least r times in θ and has not yet been used in σ_r, execute v_j. We claim that according to this construction, segment σ_r $(1 \leqslant r \leqslant n)$ uses exactly one occurrence of each rule that appears at least r times in θ. Suppose, to the contrary, that at some point in the construction of σ_r, there are no enabled rules in the nonempty set S of rules that appear at least r times in θ but which have not been used in σ_r. Without loss of generality, assume σ_r is the first segment for which this happens. Let v_j be the first rule used in σ that also appears in S, and let w be the vector produced by the first r segments of σ. Now there must exist an i, $1 \leqslant i \leqslant k$, such that $w(i) = 0$ and $U(i, j) = 1$. If $V(i, j) \neq -1$, then from the definition of conflict-freedom, no rule can subtract from position i, so position i would have had to have been 0 throughout σ. But this would mean v_j could not have been executed even once—a contradiction. Therefore, $V(i, j) = -1$. Since $\delta(\theta) + v_0 \geqslant \mathbf{0}$, some rule $v_{j'}$ used in σ_r or occurring in S must add to position i. Since v_j is the only rule that can subtract from position i, $v_{j'}$ cannot have been executed since the last time v_j was executed; otherwise, v_j would be enabled by w. Thus, $v_{j'} \in S$. Now clearly, v_j and $v_{j'}$ have been executed the same number of times in the first r segments, so $v_0(i) \leqslant w(i) = 0$. Since v_j is the first rule from S used in σ, some other rule (not $v_{j'}$) which adds to position i must have been executed before v_j was first used. But this forces $w(i) > 0$—a contradiction. Therefore, segment σ_r contains exactly one occurrence of each rule that appears at least r times in θ, for $1 \leqslant r \leqslant n$. Thus, the sequence θ' of rules used in σ satisfies the lemma. ∎

The following is a corollary to the proof of Lemma 3.2; it will be used in obtaining later results.

Corollary 3.1. *If θ is a sequence of rules enabled at v such that $\delta(\theta) \geqslant \mathbf{0}$, then there exists a vector $v' \leqslant v$ with no element larger than 1 and a sequence of rules θ' with $\Psi(\theta) = \Psi(\theta')$ such that θ' is enabled at v'.*

Proof. Let $v'(i) = 0$ if $v(i) = 0$, $v'(i) = 1$ otherwise. Consider the first segment constructed in the proof of Lemma 3.2. Since each rule used in θ is used exactly once in this segment, no more than 1 is subtracted from any position during the execution of the segment. Thus, this segment is clearly enabled by v'. Now from Lemma 3.2, there is some sequence θ' enabled at v' such that $\Psi(\theta) = \Psi(\theta')$. ∎

We are now ready to show the reachability problem to be NP-complete. Recall that the problem was shown to be NP-hard in [15]. An inspection of the construction used in that proof reveals that it holds for both conflict-free Petri nets and conflict-free VASs. Hence, we only need to show the upper bound.

Theorem 3.1. *The reachability problem for conflict-free VRSs is NP-complete.*

Proof. Let (v_0, U, V) be a $k \times m$ conflict-free VRS, and let $w \in N^k$ be an arbitrary vector. Our algorithm assumes the existence of some path that results in w, and guesses the set of rules used in that path. It then verifies whether there is some path which uses exactly this set of rules. By Lemma 3.1, this can be verified in polynomial time. Let the set of guessed rules be the $k \times n$ matrix V'. Our algorithm now verifies that there is some $x \in N^n$, $x(i) \geqslant 1$ for $1 \leqslant i \leqslant n$, such that $V'x + v_0 = w$. From Borosh and Treybig [2], this can be verified in NP. Now from Lemma 3.2, if such an x exists, then $w \in R(v_0, U, V)$. ∎

4. The Containment and Equivalence Problems

We now turn to the containment and equivalence problems. We will show that these problems are $\prod_2^P$-complete, where $\prod_2^P$ is the set of complements of all languages that can be recognized by a polynomial-time-bounded nondeterministic Turing machine with an NP oracle (see Stockmeyer [28]). A part of our proof is derived from a technique used first by Huynh [14] (see also [10]). In [14], Huynh gave a proof that the containment and equivalence problems for semilinear sets are in $\prod_2^P$. Landweber and Robertson [18] have shown that the reachability set of a conflict-free Petri net is semilinear; it is easy to verify that this also holds for VRSs. In what follows, we give an upper bound on the size of the SLS representation of the reachability set. In particular, we give an SLS representation in which no integer is larger than $(c*k*m*n)^{d*k*m}$, where k and m are the dimensions of the VRS, n is the largest absolute value of any integer in the VRS, and c and d are fixed constants independent of k, m, and n. Now the technique used in [14] is to show that if the two SLSs are not equal, then there is a "small" witness to that fact. Unfortunately, applying our derived bounds to the result in [14] yields a bound of $O((k*m*n)^{(k*m*n)^{c'*k^2*m}})$ for the largest integer in the smallest witness. This is clearly too large to guess in polynomial time. Furthermore, our bounds cannot be improved enough to make a direct application of Huynh's results work. To see this, first note that Huynh's upper bound is in terms of the number of periods in the SLSs, the dimension of the vectors, the maximum integer in either SLS, and the total number of linear sets. Now consider, for arbitrary n and k, the $(k+1) \times k$ conflict-free VAS with start vector $(1, 0, ..., 0)$ and the following addition rules:

- for each position i, $2 \leqslant i \leqslant k$, a rule which will decrement position $i-1$ and add n to position i; and
- a rule which adds n to position $k+1$.

It is not hard to see that any SLS representation for the above VAS must have at least n^k linear sets, n^k periods, and a maximum integer of at least n^k. Now even if this example represented the worst case, Huynh's results yield a bound of $O((k*n)^{(k*n)^{c'*k}})$. In [10], a variation of the proof in [14] was given in which a small enough bound was placed on the sizes of the periods to allow some degree of improvement to be made. Now in the above example, any SLS representation must clearly contain periods with integers at least n. Unfortunately, even if a bound of n could be placed on the largest integer in any period, this proof does not yield a polynomial bound on the binary representation of the smallest witness.

What we are able to do, however, is to give a SLS representation with a high degree of symmetry. We then consider two cases. In one case, we are able to make use of some of the techniques in [14] to give a tight bound on the size of the smallest witness. In the other case, the symmetry of the SLS allows us to show the existence of a witness without actually having to exhibit it. The following lemma gives the SLS representation of the reachability set of an arbitrary conflict-free VRS.

LEMMA 4.1. *Let (v_0, U, V) be a $k \times m$ conflict-free VRS in which n is the largest absolute value of any integer. Then there exist constants c_1, c_2, d_1, and d_2, independent of k, m, and n, such that $R(v_0, U, V) = \bigcup_{v \in B} \mathscr{L}(v, P_v)$, where B is the set of all reachable vectors with no element larger than $(c_1^* k^* m^* n)^{c_2 * k * m}$, and P_v is the set of all displacements of positive loops enabled at v such that if $p \in P_v$, then*

1. *p has no element larger than $(d_1^* k^* m^* n)^{d_2 * k * m}$, and*
2. *if $v(i) = 0$, then $p(i) = 0$.*

Proof. $(\bigcup_{v \in B} \mathscr{L}(v, P_v) \subseteq R(v_0, U, V))$. Clear.

$(R(v_0, U, V) \subseteq \bigcup_{v \in B} \mathscr{L}(v, P_v))$. Let $y \in R(v_0, U, V)$. Then there is a sequence θ_1 enabled at v_0 such that $v_0 + V\Psi(\theta_1) = y$. Let w_1 be a k-dimensional vector such that $w_1(i) = 0$ iff $y(i) = 0$, $w_1(i) = 1$ otherwise, and let w_2 be an m-dimensional vector such that $w_2(j) = 0$ iff $\Psi(\theta_1)(j) = 0$, and $w_2(j) = 1$ otherwise. Consider the following system of linear Diophantine inequalities:

$$
\begin{aligned}
& v_0 + Vx \geqslant w_1 \\
& v_0 + Vx = v \\
& \qquad x(j) = 0 \qquad \text{iff} \quad w_2(j) = 0 \\
& \qquad x(j) \geqslant 1 \qquad \text{iff} \quad w_2(j) = 1.
\end{aligned}
$$

Clearly, $\langle \Psi(\theta_1), y \rangle$ is a solution of the above system over the variables $\langle x(1), ..., x(m), v(1), ..., v(k) \rangle$. Furthermore, from Lemma 3.2, *any* solution to the above system is equivalent to some pair $\langle \Psi(\theta_2), w \rangle$, where θ_2 is a sequence enabled at v_0 which yields w when executed at v_0. Let $\langle \Psi(\theta_2), w \rangle$ be a minimal solution such that $\langle \Psi(\theta_2), w \rangle \leqslant \langle \Psi(\theta_1), y \rangle$. From results of von zur Gathen and Sieveking [31] and Huynh [12], there exist constants c_1 and c_2 such that no element of w is larger than $(c_1^* k^* m^* n)^{c_2 * k * m}$. Note that since we can assume without loss of generality that $n \geqslant 1$, c_1 and c_2 are independent of w_1 and w_2, and hence of y. If we now assign the values of c_1 and c_2 to the constants (of the same name) in the definition of B given in the statement of the lemma, then $w \in B$.

We will now show that $y \in \mathscr{L}(w, P_w)$. Since $\mathscr{V}$ is conflict-free and $\Psi^+(\theta_1) \geqslant \Psi^+(\theta_2)$, it is easy to see that there is a sequence θ_3 enabled at w such that $\Psi^+(\theta_1) = \Psi^+(\theta_2) + \Psi^+(\theta_3)$. Since $w_1 \leqslant w \leqslant y$, if $w(i) = 0$, then $\delta(\theta_3)(i) = 0$. Clearly, $\Psi^+(\theta_3) \geqslant \mathbf{0}$. Let w_3 be an m-dimensional vector such that $w_3(j) = 0$ iff

$\Psi(\theta_3)(j)=0$ and $w_3(j)=1$ otherwise. Consider the following system of linear Diophantine inequalities:

$$\begin{aligned} &Vx \geqslant \mathbf{0} && \\ &Vx = v && \\ &x(j)=0 && \text{iff} \quad w_3(j)=0 \\ &x(j)\geqslant 1 && \text{iff} \quad w_3(j)=1. \end{aligned}$$

Clearly, $\Psi^+(\theta_3)$ is a solution to the above system over the variables $\langle x(1), ..., x(m), v(1), ..., v(k)\rangle$. Furthermore, from Lemma 3.2, *any* solution is an extended Parikh map of a loop enabled at w. Let $\Psi^+(\theta_4)$ be a minimal solution of the above system such that $\Psi^+(\theta_4)\leqslant \Psi^+(\theta_3)$. As above, no element in $\delta(\theta_4)$ is larger than $(d_1{*}k{*}m{*}n)^{d_2*k*m}$, where d_1 and d_2 are constants independent of k, m, n, w, and y. We now assign the values of d_1 and d_2 to the constants (of the same name) in the definition of P_v given in the statement of the lemma. Notice that $\mathbf{0}\leqslant \Psi^+(\theta_3)-\Psi^+(\theta_4)<\Psi^+(\theta_3)$ if $\Psi^+(\theta_3)\neq \mathbf{0}$. Since θ_4 is enabled at w and $\mathscr{V}$ is conflict-free, there must be some loop enabled at $w+\delta(\theta_4)$ whose Parikh map is $\Psi(\theta_3)-\Psi(\theta_4)$. Furthermore, since $w_1\leqslant w\leqslant w+\delta(\theta_4)$, from Corollary 3.1, this new loop is enabled at w_1, and hence at w. Thus, the above procedure may be iterated, breaking θ_3 into loops satisfying the definition of P_w. Therefore, $y\in\mathscr{L}(w, P_w)$, and hence $y\in\bigcup_{v\in B}\mathscr{L}(v, P_v)$. ∎

The key feature of the SLSs given in the above lemma is that they will allow us to consider only two sets of periods—one set from each of the two VRSs under consideration. In particular, suppose we want to establish that $R(\mathscr{V}_1)\not\subseteq R(\mathscr{V}_2)$; i.e., we wish to illustrate the existence of a $w\in SL_1\backslash SL_2$, where SL_1 and SL_2 are the corresponding semilinear sets given in Lemma 4.1. Let B_1 (B_2) be the set defined as B in Lemma 4.1 with respect to $\mathscr{V}_1$ ($\mathscr{V}_2$, respectively), and for an arbitrary vector v, let P^1_v (P^2_v) denote the set defined as P_v in Lemma 4.1 with respect to $\mathscr{V}_1$ ($\mathscr{V}_2$, respectively). Suppose we have a $w\in SL_1$. Then $w\in\mathscr{L}(b_1, P^1_{b_1})$ for some $b_1\in B_1$. If, in addition, $w\in SL_2$, then $w\in\mathscr{L}(b_2, P^2_{b_2})$ for some $b_2\in B_2$. We will show in Lemma 4.4 that $P^2_{b_1}=P^2_{b_2}$. Thus, if $w\notin SL_2$, we can verify this by showing that $w\notin\bigcup_{v\in B'_2}\mathscr{L}(v, P^2_{b_1})$, for some $B'_2\subseteq B_2$. (We will show in Lemma 4.4 how to construct B'_2.) So to illustrate the existence of a $w\in SL_1\backslash SL_2$, it is sufficient to illustrate the existence of a $w\in\mathscr{L}(b_1, P^1_{b_1})\backslash\bigcup_{b\in B'_2}\mathscr{L}(b, P^2_{b_1})$ for some $b_1\in B_1$ and a particular $B'_2\subseteq B_2$. Note that in this process we are only concerned with two period sets, $P^1_{b_1}$ and $P^2_{b_1}$.

Consider two sets, $\mathscr{L}(b_1, P_1)$, and $\bigcup_{b\in B}\mathscr{L}(b, P_2)$. Furthermore, suppose that every vector in P_1 is a positive linear combination of the vectors in P_2. In Lemma 4.2, we use some of the techniques given in [14] to show that if $\mathscr{L}(b_1, P_1)$ is not contained in $\bigcup_{b\in B}\mathscr{L}(b, P_2)$, there must be a "small" witness to this fact. In particular, we will show the existence of a witness whose largest element is linear in the size of the largest element in the representations of P_1, P_2, B, and b_1, and exponential in the dimension. By applying this bound to the bounds given in

Lemma 4.1, we will have shown the existence of a witness that can be written down in space polynomial in the description of the original VRSs. On the other hand, suppose some period in P_1 is not a positive linear combination of the vectors in P_2. We will show in Lemma 4.3 that in this case, $\mathscr{L}(b_1, P_1)$ cannot be contained in $\bigcup_{b \in B} \mathscr{L}(b, P_2)$.

LEMMA 4.2. *Let P_1, P_2, and B be finite subsets of N^k, $b_1 \in N^k$, and $n \in N$ such that no integer in P_1, P_2, B, or b_1 exceeds n. If every vector in P_1 is a positive linear combination of vectors in P_2 and $w \in \mathscr{L}(b_1, P_1) \backslash \bigcup_{b \in B} \mathscr{L}(b, P_2)$, then there is a w' with no element larger than $k^*n^{2^*k+1} + n$ such that $w' \in \mathscr{L}(b_1, P_1) \backslash \bigcup_{b \in B} \mathscr{L}(b, P_2)$.*

Proof. Suppose w contains some element larger than $k^*n^{2k+1} + n$. We will show that there is a $w' < w$ in $\mathscr{L}(b_1, P_1) \backslash \bigcup_{b \in B} \mathscr{L}(b, P_2)$. Let $P_1 = \{p_1, ..., p_m\}$, and let $w = b_1 + \sum_{j=1}^{m} a_j^* p_j$, where each $a_j \in N$. Since b_1 has no component larger than n, $\sum_{j=1}^{m} a_j^* p_j$ contains some component larger than k^*n^{2k+1}. By the pigeonhole principle, there is some $a_h > k^*n^k$. Since p_h is a positive linear combination of vectors from P_2, from Caratheodory's theorem for cones (see, e.g., [29]), there is a matrix A with no more than k linearly independent columns, each of which is an element of P_2, such that $Ax = p_h$ has a unique nonnegative solution. Furthermore, if A is not square, linearly dependent rows may be removed from A and p_h yielding the system $A'x = p'_h$ with the *same* unique solution and a square matrix A'. From Cramer's rule, $x(j) = \det(A'_{[j]})/\det(A')$, where $A'_{[j]}$ is obtained by replacing the jth column of A' with p'_h. If we now let $\lambda = |\det(A')|$, $Ax = \lambda p_h$ has a nonnegative integer solution. Furthermore, $\lambda \leqslant k^*n^k < a_h$. Hence, $w' = b_1 - \lambda p_h + \sum_{p_j \in P_1} a_j^* p_j$ is in $\mathscr{L}(b_1, P_1)$ and $w' < w$. However, $w' \notin \bigcup_{b \in B} \mathscr{L}(b, P_2)$; otherwise, $w = w' + \lambda^* p_h = w' + Ax \in \bigcup_{b \in B} \mathscr{L}(b, P_2)$. ∎

LEMMA 4.3. *Let p, $b_1 \in N^k$ such that $p \neq \mathbf{0}$, and let P and B be finite subsets of N^k. If p is not a positive linear combination of the vectors in P, then there is an $n \in N$ such that $b_1 + n^*p \notin \bigcup_{b \in B} \mathscr{L}(b, P)$.*

Proof. Let $P = \{p_1, ..., p_m\}$, and let x be a nonnegative element of R^m that minimizes $\max_{1 \leqslant i \leqslant k}\{|p(i) - \sum_{j=1}^{m} p_j(i)^* x(j)|\} = \Delta$. Since p is not a positive linear combination of the vectors in P and $p \neq \mathbf{0}$, $\Delta > 0$. Let $n \in N$, be such that $n > \max_{1 \leqslant i \leqslant k, b \in B}\{|b(i) - b_1(i)|\}/\Delta$. We claim that $b_1 + n^*p \notin \bigcup_{b \in B} \mathscr{L}(b, P)$. Suppose, on the contrary, that there exist $b \in B$ and $y \in N^m$ such that $b_1 + n^*p = b + \sum_{j=1}^{m} p_j^* y(j)$. Then $p - \sum_{j=1}^{m} p_j^* y(j)/n = (b - b_1)/n$, and for all $1 \leqslant i \leqslant k$, $|p(i) - \sum_{j=1}^{m} p_j(i)^* y(j)/n| = |(b(i) - b_1(i))/n| < \Delta$. But this contradicts the choice of x. Thus, $b_1 + n^*p \notin \bigcup_{b \in B} \mathscr{L}(b, P)$. ∎

We are now ready to show our upper bound for the containment and equivalence problems for conflict-free VRSs.

LEMMA 4.4. *The containment and equivalence problems for conflict-free VRSs are in $\prod_2^P$.*

Proof. Recall that $\prod_2^P$ is the set of all complements of languages that can be recognized by a polynomial-time-bounded nondeterministic Turing machine with an NP oracle. In what follows, we motivate and describe such an algorithm for noncontainment. The fact that a similar algorithm works for inequivalence will subsequently be obvious.

Let $\mathscr{V}_1$ and $\mathscr{V}_2$ be $k \times m$ conflict-free VRSs such that the largest absolute value of any integer in either VRS is n. We wish to establish whether there is a $w \in R(\mathscr{V}_1) \backslash R(\mathscr{V}_2)$. From Lemma 4.1, if w exists, it must be in some linear set $\mathscr{L}(b_1, P^1_{b_1})$, where $b_1 \in R(\mathscr{V}_1)$ and b_1 and $P^1_{b_1}$ are as given in the lemma. Also from Lemma 4.1, $R(\mathscr{V}_2) = \bigcup_{v \in B} \mathscr{L}(v, P^2_v)$, where B and P^2_v are again as given in the lemma. Suppose $b_1 \in R(\mathscr{V}_2)$. Without loss of generality, assume $b_1 \in B$, and consider the set $P^2_{b_1}$. If $x \in \mathscr{L}(b_1, P^1_{b_1}) \cap R(\mathscr{V}_2)$, it must be in some linear set $\mathscr{L}(b_2, P^2_{b_2})$, where $b_2 \in B$. From the properties of $P^1_{b_1}$ and $P^2_{b_2}$, $x(i) = 0$ iff $b_1(i) = 0$ iff $b_2(i) = 0$ for $1 \leqslant i \leqslant k$. From Corollary 3.1, for any loop enabled at b_1, there is a loop with the same displacement enabled at some $y \leqslant b_1$, where no component of y is greater than 1. Clearly, $y \leqslant b_2$, so the same loop is enabled at b_2. Hence, $P^2_{b_2} = P^2_{b_1}$ (see the definition in Lemma 4.1). In order to show $w \in R(\mathscr{V}_1) \backslash R(\mathscr{V}_2)$, it is therefore sufficient to show $w \in \mathscr{L}(b_1, P^1_{b_1}) \backslash \bigcup_{v \in B'} \mathscr{L}(v, P^2_{b_1})$, where $B' = \{v\colon v \in B \text{ and } v(i) = 0 \text{ iff } b_1(i) = 0 \text{ for } 1 \leqslant i \leqslant k\}$.

Suppose every vector in $P^1_{b_1}$ is a positive linear combination of vectors from $P^2_{b_1}$. Then from Lemma 4.2, if there is a $w \in \mathscr{L}(b_1, P^1_{b_1}) \backslash \bigcup_{v \in B'} \mathscr{L}(v, P^2_{b_1})$, there is one that can be written down in a polynomial number of bits. Furthermore, b_1 and any element in $P^1_{b_1}$ or $P^2_{b_1}$ can be written down in a polynomial number of bits. Suppose, on the other hand, that there is a $p \in P^1_{b_1}$ that is not a positive linear combination of the periods in $P^2_{b_1}$. From the definition of $P^1_{b_1}$, $p \neq \mathbf{0}$; therefore, from Lemma 4.3, there is a $w \in \mathscr{L}(b_1, P^1_{b_1}) \backslash \bigcup_{v \in B'} \mathscr{L}(v, P^2_{b_1})$. From Caratheodory's theorem for cones (see, e.g., [29]), if p is a positive linear combination of vectors from a given subset P of N^k, then there is a subset $P' \subseteq P$ containing at most k vectors such that p is a positive linear combination of the vectors in P'. Thus, it can clearly be decided in NP whether a given $p \in N^k$ is a positive linear combination of vectors from $P^2_{b_1}$.

So to verify that there is a $w \in R(\mathscr{V}_1) \backslash R(\mathscr{V}_2)$, one needs only to verify that one of two situations occurs: either (1) there is a $w \in R(\mathscr{V}_1) \backslash R(\mathscr{V}_2)$ that can be guessed directly using only a polynomial number of bits, or (2) there is a $b_1 \in R(\mathscr{V}_1)$ subject to the size bounds for elements of the set B given in Lemma 4.1 with respect to $\mathscr{V}_1$, and a $p \in P^1_{b_1}$ that is not a positive linear combination of the periods in $P^2_{b_1}$. We therefore have the following algorithm:

input $\mathscr{V}_1, \mathscr{V}_2$;
either {Guess which case we have.}
  {Assume we have a small witness.}
  guess w subject to the size constraints mentioned in Lemmas 4.1 and 4.2;
  verify that $w \in R(\mathscr{V}_1)$;
  if $w \in R(\mathscr{V}_2)$ {oracle call}
    then reject
    else accept

or {other case}
  guess b_1 and p consistent with the size constraints mentioned in Lemma 4.1;
  verify that $b_1 \in R(\mathscr{V}_1)$;
  verify that $p \in P^1_{b_1}$;
  if p is a positive linear combination of $P^2_{b_1}$ {oracle call}
    then reject
    else accept

The above algorithm clearly runs in polynomial time, and from the comments made above, can accept its input iff $R(\mathscr{V}_1) \not\subseteq R(\mathscr{V}_2)$. Clearly, a similar strategy may be used to decide inequivalence. ∎

We are now ready to show our main result, that the containment and equivalence problems are $\prod_2^p$-complete. Before formally proving the theorem, we will briefly explain the strategy for showing the lower bound. Let X and Y be disjoint sets of Boolean variables, and let $F(X, Y)$ be a Boolean expression in 3DNF. Stockmeyer [28] showed the problem of deciding whether $(\forall X)(\exists Y)$: $F(X, Y) = 0$ is $\prod_2^p$-complete (the notations $(\forall X)$ and $(\exists Y)$ denote $(\forall x_1 \cdots \forall x_{n_1})$ and $(\exists y_1 \cdots \exists y_{n_2})$, respectively, where $X = \{x_1, ..., x_{n_1}\}$ and $Y = \{y_1, ..., y_{n_2}\}$). We will reduce this problem to the containment and equivalence problems. The reduction will consist of constructing two conflict-free VRSs, $\mathscr{V}_1$ and $\mathscr{V}_2$, which are identical except that $\mathscr{V}_2$ has one additional rule. Let us say that a clause in $F(X, Y)$ is *killed* if one of its literals has a value of 0. The function of the VRSs is to simulate an assignment of values to the variables in $X \cup Y$, signifying killed clauses by incrementing certain positions. The additional rule in $\mathscr{V}_2$ will allow it to kill all clauses after a complete assignment is made. Thus, if we record which clauses were killed by assignments to X variables, $R(\mathscr{V}_1) = R(\mathscr{V}_2)$ iff for any assignment of values to X there is an assignment of values to Y that results in killing all clauses. Now the VRSs must be able to record which variables have been assigned values, which clauses have been killed, and which clauses have been killed by X variables. We also wish to make our proof general enough to work for conflict-free VASs and Petri nets as well. To accommodate each of these requirements, we use two positions for each variable and eight positions for each clause.

THEOREM 4.1. *The equivalence and containment problems for conflict-free VRSs are $\prod_2^p$-complete.*

Proof. We need only show that the containment and equivalence problems are $\prod_2^p$-hard. Let $X = \{x_1, ..., x_{n_1}\}$, $Y = \{y_1, ..., y_{n_2}\}$, $X \cap Y = \varnothing$, $F(X, Y) = C_1 \vee \cdots \vee C_m$, $C_j = \alpha_{1,j} \wedge \alpha_{2,j} \wedge \alpha_{3,j}$, $\alpha_{i,j} \in \{x, \bar{x}\colon x \in X \cup Y\}$. We will define a $(2n_1 + 2n_2 + 8m) \times (3n_1 + 3n_2 + 8m)$ conflict-free VRS $\mathscr{V}_1$ and a $(2n_1 + 2n_2 + 8m) \times (3n_1 + 3n_2 + 8m + 1)$ conflict-free VRS $\mathscr{V}_2$ such that $R(\mathscr{V}_1) = R(\mathscr{V}_2)$ iff $(\forall X)(\exists Y)$: $F(X, Y) = 0$. The construction will be such that $R(\mathscr{V}_1) \subseteq R(\mathscr{V}_2)$; hence, it will also be the case that $R(\mathscr{V}_2) \subseteq R(\mathscr{V}_1)$ iff $(\forall X)(\exists Y)$: $F(X, Y) = 0$. For ease of illustration, we will treat the reachable vectors as a set of assignments to a set of variables. The addition rules will then operate on these variables. The variables we will use are $\{a_i, \bar{a}_i: 1 \leqslant i \leqslant n_1\} \cup \{b_i, \bar{b}_i: 1 \leqslant i \leqslant n_2\} \cup \{c_{i,j}, \bar{c}_{i,j}: 0 \leqslant i \leqslant 3, 1 \leqslant j \leqslant m\}$. a_i and $\bar{a}_i$ will correspond to x_i, b_i and $\bar{b}_i$ will correspond to y_i, $c_{0,i}$ and $\bar{c}_{0,i}$ will correspond to C_i, and $c_{i,j}$ and $\bar{c}_{i,j}$ will correspond to $\alpha_{i,j}$. Both $\mathscr{V}_1$ and $\mathscr{V}_2$ will have start vectors of $\mathbf{0}$.

$\mathscr{V}_1$ and $\mathscr{V}_2$ will both have the following rules:

- v_i^1, $1 \leqslant i \leqslant n_1$:

$$a_i \longleftarrow a_i + 1$$
$$c_{0,j} \longleftarrow c_{0,j} + 1 \quad \forall j \text{ for which } x_i \in C_j,\ 1 \leqslant j \leqslant m;$$

- v_i^2, $1 \leqslant i \leqslant n_1$:

$$a_i \longleftarrow a_i + 1$$
$$c_{0,j} \longleftarrow c_{0,j} + 1 \quad \forall j \text{ for which } \bar{x}_i \in C_j,\ 1 \leqslant j \leqslant m;$$

- v_i^3, $1 \leqslant i \leqslant n_2$:

$$b_i \longleftarrow b_i + 1$$
$$c_{j,k} \longleftarrow c_{j,k} + 1 \quad \forall j, k \text{ for which } \alpha_{j,k} = y_i,\ 1 \leqslant j \leqslant 3,\ 1 \leqslant k \leqslant m;$$

- v_i^4, $1 \leqslant i \leqslant n_2$:

$$b_i \longleftarrow b_i + 1$$
$$c_{j,k} \longleftarrow c_{j,k} + 1 \quad \forall j, k \text{ for which } \alpha_{j,k} = \bar{y}_i,\ 1 \leqslant j \leqslant 3,\ 1 \leqslant k \leqslant m$$

- $v_{i,j}^5$, $0 \leqslant i \leqslant 3$, $1 \leqslant j \leqslant m$:

$$c_{i,j} \longleftarrow c_{i,j} - 1$$
$$\bar{c}_{i,j} \longleftarrow \bar{c}_{i,j} + 1;$$

- $v_{i,j}^6$, $1 \leqslant i \leqslant 3$, $1 \leqslant j \leqslant m$:

$$c_{i,j} \longleftarrow c_{i,j} + 1$$
$$c_{i-1,j} \longleftarrow c_{i-1,j} + 1$$
$$\bar{c}_{i-1,j} \longleftarrow \bar{c}_{i-1,j} - 1;$$

- v_i^7, $1 \leqslant i \leqslant m$:

$$c_{1,i} \longleftarrow c_{1,i} + 1$$
$$c_{3,i} \longleftarrow c_{3,i} + 1$$
$$\bar{c}_{3,1} \longleftarrow \bar{c}_{3,i} - 1;$$

- v_i^8, $1 \leqslant i \leqslant n_1$:

$$a_i \longleftarrow a_i - 1$$

$$\bar{a}_i \longleftarrow \bar{a}_i + 1;$$

- v_i^9, $1 \leqslant i \leqslant n_2$:

$$b_i \longleftarrow b_i - 1$$

$$\bar{b}_i \longleftarrow \bar{b}_i + 1.$$

In addition to the above rules, $\mathscr{V}_2$ has the rule:

- v^{10}:

$$\begin{aligned} a_i &\longleftarrow a_i + 1, && \forall 1 \leqslant i \leqslant n_1 \\ \bar{a}_i &\longleftarrow \bar{a}_i - 1, && \forall 1 \leqslant i \leqslant n_1 \\ b_i &\longleftarrow b_i + 1, && \forall 1 \leqslant i \leqslant n_2 \\ \bar{b}_i &\longleftarrow \bar{b}_i - 1, && \forall 1 \leqslant i \leqslant n_2 \\ c_{i,j} &\longleftarrow c_{i,j} + 1, && \forall 1 \leqslant i \leqslant 3, 1 \leqslant j \leqslant m. \end{aligned}$$

Clearly, both systems are conflict-free, and $R(\mathscr{V}_1) \subseteq R(\mathscr{V}_2)$. We will call all rules *superscripted* with *i type i* rules. The type 1 rules correspond to assignments of 0 to X variables, and type 2 rules correspond to assignments of 1 to X variables. Similarly, type 3 rules correspond to assignments of 0 to Y variables, and type 4 rules correspond to assignments of 1 to Y variables. Note that the execution of a type 1 or 2 rule that corresponds with an assignment that kills clause C_j will increment $c_{0,j}$. Likewise, the execution of a type 3 or 4 rule that corresponds with an assignment that makes $\alpha_{i,j} = 0$ will increment $c_{i,j}$. Thus, the function of the types 5, 6, and 7 rules is to allow $c_{i,j}$, $1 \leqslant i \leqslant 3$, to reach any positive value if clause C_j is killed. Finally, the types 8 and 9 rules will enable rule v^{10} in $\mathscr{V}_2$, which in turn will allow $c_{i,j}$, $1 \leqslant i \leqslant 3$, to reach any positive value if all a_ks and b_ks have been incremented at least once.

Based on the above comments, we now make the following observations:

1. $a_i + \bar{a}_i$ reflects the number of value assignments made to x_i (where any assignment may be made 0, 1, or more times).

2. $b_i + \bar{b}_i$ reflects the number of value assignments made to y_i.

3. $c_{0,j} + \bar{c}_{0,j}$ reflects the number of times clause C_j has been killed by assignments to variables in X.

4. In $\mathscr{V}_1$, $c_{i,j} + \bar{c}_{i,j}$, $1 \leqslant i \leqslant 3$, can become positive only if clause C_j is killed.

5. In $\mathscr{V}_2$, $c_{i,j} + \bar{c}_{i,j}$, $1 \leqslant i \leqslant 3$, can become positive only if either clause C_j is killed or every variable in $\{X \cup Y\}$ has been assigned a value at least once.

We are now ready to show that $R(\mathscr{V}_2) \subseteq R(\mathscr{V}_1)$ iff $(\forall X)(\exists Y)$: $F(X, Y) = 0$.

($\Rightarrow$) Assume $R(\mathscr{V}_2) \subseteq R(\mathscr{V}_1)$. Let B: $X \mapsto \{0, 1\}$ be any assignment of Boolean values to the variables in X. We will show that there is a B': $Y \mapsto \{0, 1\}$ such that $F(B(X), B'(Y)) = 0$. We will first construct a path σ in $\mathscr{V}_2$. Clearly, all rules of types 1–4 are always enabled; therefore, we first execute, for each i, $1 \leqslant i \leqslant n_1$, v_i^1 if $B(x_i) = 0$, or v_i^2 if $B(x_i) = 1$. Next, we execute, for each i, $1 \leqslant i \leqslant n_2$, either v_i^3 or v_i^4. At this time, $a_i = 1$ and $b_j = 1$ for $1 \leqslant i \leqslant n_1$, $1 \leqslant j \leqslant n_2$. Thus, we can now execute, for each i, $1 \leqslant i \leqslant n_1$, and each j, $1 \leqslant j \leqslant n_2$, v_i^8 and v_j^9. Since now $\bar{a}_i = 1$ and $\bar{b}_j = 1$, $1 \leqslant i \leqslant n_1$, $1 \leqslant j \leqslant n_2$, we can execute v^{10}. Note that this leaves $a_i = 1$, $\bar{a}_i = 0$, $b_j = 1$, $\bar{b}_j = 0$, $\bar{c}_{0,l} = 0$, $1 \leqslant c_{k,l} \leqslant 2$, and $\bar{c}_{k,l} = 0$, for all $1 \leqslant i \leqslant n_1$, $1 \leqslant j \leqslant n_2$, $1 \leqslant k \leqslant 3$, $1 \leqslant l \leqslant m$. $c_{0,l}$ will be nonzero iff the assignment B kills clause C_l $(1 \leqslant l \leqslant m)$. Call the resulting vector w.

Now since $w \in R(\mathscr{V}_2)$, $w \in R(\mathscr{V}_1)$ also. Let σ' be a path to w in $\mathscr{V}_1$. Since $a_i = 1$ and $\bar{a}_i = 0$ in w, σ' must contain exactly one occurrence of either v_i^1 or v_i^2, but not both. Furthermore, the rules of types 1 and 2 must clearly produce the same values in all the $c_{0,i}$s as those in w, $1 \leqslant i \leqslant m$. Now the remaining rules in σ must make all positions $c_{i,j}$, $1 \leqslant i \leqslant 3$, $1 \leqslant j \leqslant m$, positive. If clause C_j is not killed by B, then $c_{0,j} = \bar{c}_{0,j} = 0$, and the only way for any position $c_{i,j}$, $1 \leqslant i \leqslant 3$, to become positive is for some type 3 or 4 rule to increment one of them. Since for each i, exactly one of v_i^3 or v_i^4 must be executed in σ', we can define a mapping B': $Y \mapsto \{0, 1\}$ such that $B'(y_i) = 0$ iff v_i^3 is executed in σ'. Thus, for all j, $1 \leqslant j \leqslant m$, if B does not kill C_j, then B' does. Hence, $(\forall X)(\exists Y)$: $F(X, Y) = 0$.

($\Leftarrow$) Assume $(\forall X)(\exists Y)$: $F(X, Y) = 0$. Let w be an arbitrary vector in $R(\mathscr{V}_2)$. We will show that $w \in R(\mathscr{V}_1)$. Let σ be a path to w in $R(\mathscr{V}_2)$. If σ does not use v^{10}, then clearly $w \in R(\mathscr{V}_1)$. Therefore, assume without loss of generality that σ uses v^{10}. It is clear from the proof of Lemma 3.2 that we can assume without loss of generality that some initial path σ' in σ uses exactly one occurrence of every rule used by σ. Furthermore, it is clear from the proof of Lemma 3.1 that we can assume without loss of generality that at any point in σ', the next rule to be executed is some arbitrary rule used by σ, as long as it is enabled and has not yet been executed. Now under these assumptions, before v^{10} can be executed for the first time, it must be the case that $\bar{a}_i = 1$ and $\bar{b}_i = 1$; i.e., each type 8 and type 9 rule has been executed once. Now before v_i^8 (v_i^9) can be executed, either v_i^1 or v_i^2 (v_i^3 or v_i^4) must have been executed. We will assume without loss of generality that exactly one of these two rules has been executed before v^{10} is first executed. Call the initial portion of σ ending with the first execution of v^{10} σ'', and let w' be the vector produced by σ''.

We will first show that $w' \in R(\mathscr{V}_1)$; then we will show that there is a path from w' to w in $\mathscr{V}_1$. We first execute in $\mathscr{V}_1$ the types 1 and 2 rules used in σ''. Note that since exactly one of v_i^1 and v_i^2 is used in σ'', this rule represents the assignment of a Boolean value to x_i. Let B: $X \mapsto \{0, 1\}$ represent the assignment induced by these rules. Since $(\forall X)(\exists Y)$: $F(X, Y) = 0$, there is a B': $Y \mapsto \{0, 1\}$ such that for all j, $1 \leqslant j \leqslant m$, if C_j is not killed by B, then it is killed by B'. We next execute the rules corresponding to B'. Now, the values of a_i, $\bar{a}_i$, b_j, $\bar{b}_j$, $c_{0,k}$, and $\bar{c}_{0,k}$ match their counterparts in w', $1 \leqslant i \leqslant n_1$, $1 \leqslant j \leqslant n_2$, $1 \leqslant k \leqslant m$, and for every j, $1 \leqslant j \leqslant m$, there is an i, $0 \leqslant i \leqslant 3$, such that $c_{i,j} > 0$. Furthermore, no $c_{i,j}$, $1 \leqslant i \leqslant 3$, $1 \leqslant j \leqslant m$, is greater than 1. Thus, rules of types 5–7 can clearly be used to bring the $c_{i,j}$s equal to their counterparts in w'. Hence, w' is reachable in $\mathscr{V}_1$. We can now simulate the remainder of σ as follows. We simulate σ until the next occurrence of v^{10} in σ is reached, except that we skip all occurrences of rules of types 8 and 9. (Note that the only rule in $\mathscr{V}_2$ that the type 8 and 9 rules enable is v^{10}.) Now for each subsequent occurrence of v^{10} in σ, at least one of each type 8 and type 9 rule must have previously occurred in σ. Therefore, when the simulation of σ reaches an occurrence of v^{10} (in σ), it simulates exactly one occurrence of each rule of types 8–10 using only rules of types 5–7 (as above). We continue this process until the end of σ is reached. We then execute all rules of types 8 and 9 that have not yet been simulated. Thus, it should be clear that every rule in this simulation is enabled at the proper time and that w is reached. ∎

5. Conclusions

The complexity of the reachability problem for general VRSs remains an open problem, with the best known lower bound being exponential space [19] and the best known upper bound being nonprimitive recursive [17, 22]. We have shown completeness results for the reachability, containment, and equivalence problems for conflict-free VRSs. The techniques used in this paper rely heavily on conflict-freedom. A property similar to conflict-freedom is persistence [18]. Thus, it seems quite possible that some of these techniques might extend to persistent VRSs. The complexity of the three problems for persistent VRSs is currently unknown. The three problems have a lower bound of PSPACE [27], but the best known upper bounds for the problems are nonprimitive recursive [7, 23, 25]. The two main problems in extending these techniques seem to be that persistent VRSs may contain negative numbers smaller than -1, and that the property of persistence is dependent upon the start vector. Another possible extension might be to consider an alternative definition of conflict-freedom which allows the use of numbers smaller than -1.

Acknowledgments

We would like to thank the referee for suggestions which improved the presentation of our results. We are especially thankful for suggestions that greatly simplified the upper bound proof of our main result.

References

1. H. Baker, "Rabin's Proof of the Undecidability of the Reachability Set Inclusion Problem of Vector Addition Systems," Project MAC.CSGM 79, MIT, Cambridge, MA, 1973.

2. I. Borosh and L. Treybig, Bounds on positive integral solutions of linear Diophantine equations, *Proc. Amer. Math. Soc.* **55**, No. 2 (1976), 299–304.
3. E. Cardoza, R. Lipton, and A. Meyer, Exponential space complete problems for Petri nets and commutative semigroups, *in* "Proceedings, 8th Annual ACM Symposium on Theory of Computing, 1976," pp. 50–54.
4. P. Clote, The finite containment problem for Petri nets, *Theoret. Comput. Sci.* **43** (1986), 99–106.
5. S. Crespi-Reghizzi and D. Mandrioli, A decidability theorem for a class of vector addition systems, *Inform. Process. Lett.* **3**, No. 3 (1975), 78–80.
6. A. Ginzburg and M. Yoeli, Vector addition systems and regular languages, *J. Comput. System Sci.* **20** (1980), 277–284.
7. J. Grabowski, The decidability of persistence for vector addition systems, *Inform. Process. Lett.* **11**, No. 1 (1980), 20–23.
8. M. Hack, The equality problem for vector addition systems is undecidable, *Theoret. Comput. Sci.* **2** (1976), 77–95.
9. J. Hopcroft and J. Pansiot, On the reachability problem for 5-dimensional vector addition systems, *Theoret. Comput. Sci.* **8** (1979), 135–159.
10. R. Howell, L. Rosier, D. Huynh, and H. Yen, Some complexity bounds for problems concerning finite and 2-dimensional vector addition systems with states, *Theoret. Comput. Sci.* **46** (1986), 107–140.
11. R. Howell, L. Rosier, and H. Yen, An $O(n^{1.5})$ algorithm to decide boundedness for conflict-free vector replacement systems, *Inform. Process. Lett.* **25** (1987), 27–33.
12. D. Huynh, The complexity of semilinear sets, *Electron. Informationsverarb. Kybernetik* **18** (1982), 291–338.
13. D. Huynh, The complexity of the equivalence problem for commutative semigroups and symmetric vector addition systems, *in* "Proceedings, 17th Annual ACM Symposium on Theory of Computing, 1985," pp. 405–412.
14. D. Huynh, A simple proof for the $\sum_2^p$ upper bound of the inequivalence problem for semilinear sets, *Elektron. Informationsverarb. Kybernet.* **22** (1986), 147–156.
15. N. Jones, L. Landweber, and Y. Lien, Complexity of some problems in Petri nets, *Theoret. Comput. Sci.* **4** (1977), 277–299.
16. R. Karp and R. Miller, Parallel program schemata, *J. Comput. System Sci.* **3**, No. 2 (1969), 147–195.
17. R. Kosaraju, Decidability of reachability in vector addition systems, *in* "Proceedings, 14th Annual ACM Symposium on Theory of Computing, 1982," pp. 267–280.
18. L. Landweber and E. Robertson, Properties of conflict-free and persistent Petri nets, *J. Assoc. Comput. Mach.* **25**, No. 3 (1978), 352–364.
19. R. Lipton, "The Reachability Problem Requires Exponential Space," Yale University, Dept. of CS. Report No. 62, Jan. 1976.
20. E. Mayr and A. Meyer, The complexity of the finite containment problem for Petri nets, *J. Assoc. Comput. Mach.* **28**, No. 3 (1981), 561–576.
21. E. Mayr and A. Meyer, The complexity of the word problems for commutative semigroups and polynomial ideals, *Adv. in Math.* **46** (1982), 305–329.
22. E. Mayr, An algorithm for the general Petri net reachability problem, *SIAM J. Comput.* **13**, No. 3 (1984), 441–460; preliminary version, *in* "13th Annual Symposium on Theory of Computing, 1981."
23. E. Mayr, Persistence of vector replacement systems is decidable, *Acta Inform.* **15** (1981), 309–318.
24. K. McAloon, Petri nets and large finite sets, *Theoret. Comput. Sci.* **32** (1984), 173–183.
25. H. Müller, On the reachability problem for persistent vector replacement systems, *Computing Suppl.* **3** (1981), 89–104.
26. H. Müller, Weak Petri net computers for Ackermann functions, *Elektron. Informationsverarb. Kybernet.* **21** (1985), 236–244.
27. L. Rosier and H. Yen, A multiparameter analysis of the boundedness problem for vector addition systems, *J. Comput. System Sci.* **32**, No. 1 (1986), 105–135.

28. L. STOCKMEYER, The polynomial-time hierarchy, *Theoret. Comput. Sci.* **3** (1977), 1–22.
29. J. STOER AND C. WITZGALL, "Convexity and Optimization in Finite Dimensions I," Springer-Verlag, New York/Berlin, 1970.
30. R. VALK AND G. VIDAL-NAQUET, Petri nets and regular languages, *J. Comput. System Sci.* **23** (1981), 299–325.
31. J. VON ZUR GATHEN AND M. SIEVEKING, A bound on solutions of linear integer equalities and inequalities, *Proc. Amer. Math. Soc.* **72**, No. 1 (1978), 155–158.